Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 97 (2014) 1745 – 1754

Procedia Engineering

www.elsevier.com/locate/procedia

12th GLOBAL CONGRESS ON MANUFACTURING AND MANAGEMENT, GCMM 2014

A Comparison of Artificial Bee Colony algorithm and Genetic Algorithm to minimize the makespan for Job Shop Scheduling

Muthiah A[a]* and Rajkumar R[b]

[a]*Asst.Professor, Department of Mechanical Engineering, P.S.R. Engineering College, Sivakasi – 626140, India*
[b]*Professor, Department of Mechanical Engineering, Mepco Schlenk Engineering College, Sivakasi – 626015, India*

Abstract

Job shop scheduling is predominantly an Non deterministic polynomial (NP)- complete challenge which is successfully tackled by the ABC algorithm by elucidating its convergence. The Job Shop Scheduling Problem (JSSP) is one of the most popular scheduling models existing in practice which is among the hardest combinatorial optimization problems. The ABC (Artificial Bee Colony) technique is concerned, it is observed that the entire specific artificial bees move about in a search space and select food sources by suitably adapting their location, know-how and having a full awareness of their nest inhabitants. Moreover, several scout bees soar and select the food sources discretely without making use of any skills. In the event of the quantity of the nectar in the fresh source becoming larger than the nectar quantity of an available source, they remember the fresh location and conveniently disregard the earlier position. In this way, the ABC system integrates local search techniques, executed by employed and onlooker bees, with universal search approaches, administered by onlookers and scouts. In our ambitious approach we have employed these three bees to furnish optimization in makespan, machine work load and the whole run period in an optimized method. In this way, with the efficient employment of our effective technique we make an earnest effort to minimize the makespan and number of machines. This paper compares GA to minimize the make span of the job scheduling process with ABC and proved that ABC algorithm produces the better result.

Selection and peer-review under responsibility of the Organizing Committee of GCMM 2014

Keywords: *Job Shop scheduling; make span; artificial bee colony optimization; GA*

* Corresponding author. Tel.: +91-9842596590
E-mail address: amuthiah68@gmail.com

Selection and peer-review under responsibility of the Organizing Committee of GCMM 2014
doi:10.1016/j.proeng.2014.12.326

1. Introduction

Permutation sequence-dependent setup times are the most significant combinational optimization challenges extensively encountered by the industries everywhere [1]. Of late, scheduling issues have assumed a crucial part on account of the zooming customer quest for diversity, abridged product life periods, ever-progressing markets with international competition and swift expansion of sophisticated technologies [2]. Generally, Job-shop scheduling is, in fact, a powerful NP-complete issue. Many an investigator has launched innovative JSP model in terms of issue constraints. It is essential that every job must be processed without preemption for a pre-determined time frame in respect of a particular machine [3]. A schedule, in turn, is an allotment of functions to time intervals on a machine. The makespan represents the highest run time of the processes and the aim of the JSP is to arrive at a schedule that ensures the least make span which represents the interval between the start of the initial process to the end of the final process, i.e. from start to finish. A good schedule is A schedule which achieves ensures the least idle time for the machines is considered to be a superb one [4]. Artificial neural network brands have been efficiently executed to tackle with a job-shop scheduling problem (JSSP) called a Non polynomial (NP-complete) trait contentment issue. Usually, an NP-complete issue is one which for a certain input of dimension N consumers a time in direction proportion to at least 2N [5]. Here, we follow a flow shop mechanism comprising M machines which are intended to process N similar tasks. It is to be noted that for getting swift services, added attention and administration are highly essential. On the other hand, prolonged services cause hindrances by taking longer runtimes which result in missing the time frame desired. Our target in this investigation is to arrive at the cost reducing service period [6]. And we target at reducing a certain standards, which are preset and known earlier. Every function of JSP process is capable of being processed on any machine from among a set of existing machines and the runtime on diverse machines are obviously dissimilar. In this way, JSP is endowed with the potency of decreasing the machine parameters, widening the search domain of realistic solutions, thereby tackling the challenges faced [7].

GA has been widely performed on several scheduling issues and is observed to present a superb performance in many of the cases. [8]. The GA employs two vectors to symbolize solutions. Advanced crossover and mutation operators are employed to harmonize to the extraordinary chromosome structure and the traits of the issue. The primary population is segregated into sub-populations, and every sub-population is formulated individually. Communication between sub-populations is limited to the migration of chromosomes [9]. Anyhow, the current genetic algorithms for the JSSP are generally found to have a slow convergence tempo and hence it is effortless to entrap local optimal solutions [10]. To speed up the process we employ hybridization of Genetic Algorithm along with Artificial Bee Colony algorithm. The Bee Colony is an optimization process triggered by the attitude of nectar collecting honeybees well represented by “waggle dance” [11]. In the ABC algorithm, the artificial bees are categorized into three distinct groups: the employed bees, the onlookers and the scouts. A bee that takes advantage of a food source is termed as employed. On the other hand, the onlooker enjoys the dances of the employed bees. The duties entrusted to scout bees are to hunt for fresh food resources discretely in the neighborhood of the hive [12]. This throws open before them several challenges such as being effortlessly entrapped in local optimum and in this the calculation period is found to be extremely high [13]. Moreover we resort to employing the hybridization of Genetic Algorithm with PSO, which takes cues from the scrutiny of the community nature of animals, like birds in flocks or fish in schools as well as on swarm concept. The rapport between the swarm and particles in PSO is identical to the association between the population and chromosomes in a GA [14]. The population of PSO is known by the name ’swarm’, and every individual in swarm is termed as a particle. Every particle is a prospective solution in the PSO and is identified with its present location and velocity [15].

With a view to tackle this menace, we employ ABC algorithm in our new technique which is capable of yielding amazing accuracy in addition to reducing to the least the make span and generating incessant outcomes.

In this investigation, we aim at reducing the following three parameters:

(1) The minimization of make span of the tasks.

(2) The minimization of maximal machine workload, represented by the maximum runtime consumed by any machine. The aim is to block a solution from allocating colossal tasks on a solitary machine and also to maintain equilibrium of work allocation among the machines.

(3)The minimization of the entire workload, symbolized by the overall run time allocated to all the machines. This aim has special significance, in cases where machines have diverse efficiency levels.

2. Literature Review

The yesteryears have enthusiastically witnessed the august appearance and advancement of the amazing Job-Shop Scheduling Problem (JSSP) which has invoked the interest of inquisitive investigators, primarily on account of its inherent combinatorial traits, making it very hard to solve. Triggered by the superb performance turned out by local search processes, these relentless researchers have resolved to integrate local search engines with universal technologies. Anan Banharnsakun *et.* [16] have amazingly launched an innovative and efficient scheduling technique using Best-so-far Artificial Bee Colony (Best-so-far ABC) for tackling the unresolved issues of the JSSP. They have, in their technique, inclined the solution direction in the direction of the Best-so-far solution disregarding an adjacent solution advocated in the original ABC method. They have also resorted to exploit the set theory to define the mapping of their novel approach to the vexed issue haunting the combinatorial optimization region. The efficiency of the epoch-making technique was scientifically estimated by using 62 benchmark issues extracted from the Operations Research Library (OR- Library). The solution excellence was graded based on ‘‘Best’’, ‘‘Average’’, ‘‘Standard Deviation (S.D.)’’, and ‘‘Relative Percent Error (RPE)’’of the objective value. The cheering outcomes have established without an iota of doubt that our innovative technique has turned out superb quality solutions in relation to the traditional high-tech heuristic-based algorithms.

Rui Zhang *et al*. [17] have remarkably given shape to an innovative discrete artificial bee colony (DABC) algorithm for tackling the multi-objective flexible job shop scheduling hassles with preservation tasks. The efficiency benchmarks taken into consideration include the highest run time extensively expressed as makespan, the overall workload of machines and the workload of the crucial machine. A proficient initialization technique is devised to build the primary population with a distinct echelon of excellence and multiplicity. A self-adaptive technique is followed to ensure the DABC algorithm with learning competence for yielding adjacent solutions in various potential areas whereas an exterior Pareto archive set is formulated to document the non-dominated solutions located till now. Moreover, an innovative deciphering device is introduced to take suitable care of preservation tasks in schedules produced. The projected DABC algorithm is experimented on a set of the famous yardstick examples extracted from the modern literature. An extensive and impartial assessment of the performance of our novel method well-backed by test outcomes proclaims to the world in no unclear terms the superb excellence and charismatic performance of the system.

Antonin Ponsich *et al*. [18] have proficiently put forward the novel concept of hybridizing DE (Differential Evolution) with Tabu Search (TS) with a view to tackle the issues of the JSSP. Aggressive vicinity is incorporated within the TS with a target of assessing whether DE is competent to substitute the re-start traits forming part of the core competencies of i-TSAB represented by a long-term memory and a path-re linking process. The computation investigations recorded in respect of 100 plus JSSP models have exhibited the fact that the projected hybrid DE–TS algorithm has mercilessly beaten down parallel high-tech methods. Still, we feel, there is greater scope for refinement if the satisfactoriness between the solution representation modes within DE and TS is suitably given thrust to.

In 2013, T.C.Wong *et al*. [19] has wonderfully launched a hybrid genetic algorithm (HGA) and a hybrid particle swarm optimization (HPSO) designed for tackling the issues inherent in AJSSP by taking into account the LS technique. Job shop scheduling problem (JSSP) turns a blind eye to assembly association and lot splitting. If an assembly phase is annexed to JSSP for the ultimate product, the issue then metamorphoses into an assembly job shop scheduling problem (AJSSP). To enable lot splitting, lot streaming (LS) method is considered in which jobs may be divided into a number of smaller sub-jobs for analogous processing on diverse phases so that the system efficiency takes a dynamic leap forward. In the present investigation, the system mission is demarcated as the makespan reduction to the possible. A constraint of the captioned model is the dearth of its relevance to situations wherein the lot dimension suffers either from lack of distinctness or from rigidity.

Rui Zhang *et al*. [20] have resourcefully targeted at reducing the overall weighted tardiness in JSSP. Taking into account the increased intricacy, an artificial bee colony (ABC) algorithm is launched for tackling the perennial paradox. A vicinity trait of the issue is found out, and thereafter a tree search algorithm is formulated to boost the utilization potential of ABC. On the basis of cheering outcomes of wide-ranging investigations, it is crystal clear the

well-conceived technique goes northward in terms of excellence in successfully tackling the job shop scheduling problem with total weighted tardiness benchmark.

3. Problem Definition

The sluggish job-shop scheduling problem entails processing of predetermined jobs by predetermined machines. A superior schedule must be capable of minimizing the redundant interval by which the machines are delayed. Hence, the decisive issue is to calculate the process series on the machines with a view to reduce the make span to the minimum. By make span, we mean the time interval needed from the start of the initial process to the close of the final process, i.e. from beginning to end. Every one of the job comprises a pre-fixed series of process actions, which required to be processed without causing any preemption for a scheduled period of interval on a predetermined machine. As functions of the identical job are not capable of being processed simultaneously, each job is entrusted with the task of visiting every machine precisely once. In the job shop scheduling problem (JSSP), a set of n jobs are to be processed on a set of m machines. Each job has a steady processing path which moves through all the machines in a prearranged manner. At last, the algorithm is experimented on instances of 8 working processes and 5 machines. Every solitary machine is expected to process 5 jobs and every job, in turn, must perform about 8 diverse tasks. In our novel technique, we have carried out the tests with solitary machine which has 5 diverse jobs, with each job discharging 8 diverse tasks.

3.1. Assumption in our proposed work

In our paper we have taken 5 different jobs with 8 different process and the technique is applicable with any type of job and process We consider the flexible job shop case where stages might be skipped. In our work the following assumption are being made and they are as follows.

- All 5 jobs must be processed in a single machine.
- In each job must be processed in the allocated time
- Each job can be processed with the shortest time such that it completes with shortest time.

At a given time, a machine can execute at most one operation.

4. Proposed Methodology

The Job Shop Scheduling Problem (JSSP) is an annex of the traditional job scheduling problem which enables a task to be processed by any of the machines in a prearranged set. The important function is to allocate each and every task to a machine and to organize the functions on the machines, in order that the maximum duration of completion (make span) of the entire task is reduced to the minimum. Compared with the traditional Job-shop Scheduling Problem (JSP, the flexible job-shop scheduling problem (FJSP) is an extension of the classical JSP, which is a more complex NP-hard problem. Each operation of FJSP job can be processed on any among a set of available machines and operation on different machines needs different time. Thus, FJSP reduces the machine constraints, enlarges the search range of feasible solution, and hence increases the difficulties greatly [7].

4.1. ABC algorithm

In our projected task, we are launching a new model for the decrease of make span by means of Artificial Bee Colony (ABC) algorithm, with a view to ensure that the time interval for the finish of the entire scheduling procedure is brought down to the least possible time. For this function, we are allotting n jobs and m process for a solitary machine task. With an eye on tackling prohibiting time interval and a large number of machines for the purpose of accomplishing tasks, we have employed ABC algorithm as our innovative part technique. At first, every job has its individual function which is demarcated by bee. The bee thus engendered is entrusted with the task of locating the nectar quantity of the food source relative to fitness value and in this task, if the fitness value is lower, then the survival prospect tends to be higher. In this regard, run interval and machines allocations are the most important constraints. In our envisaged work, we have targeted our aim at decreasing the overall makespan, solitary machine work load and the overall work load. The trigger of this investigation is to execute ABC algorithms on the modules level so as to achieve advantage from its vigor.

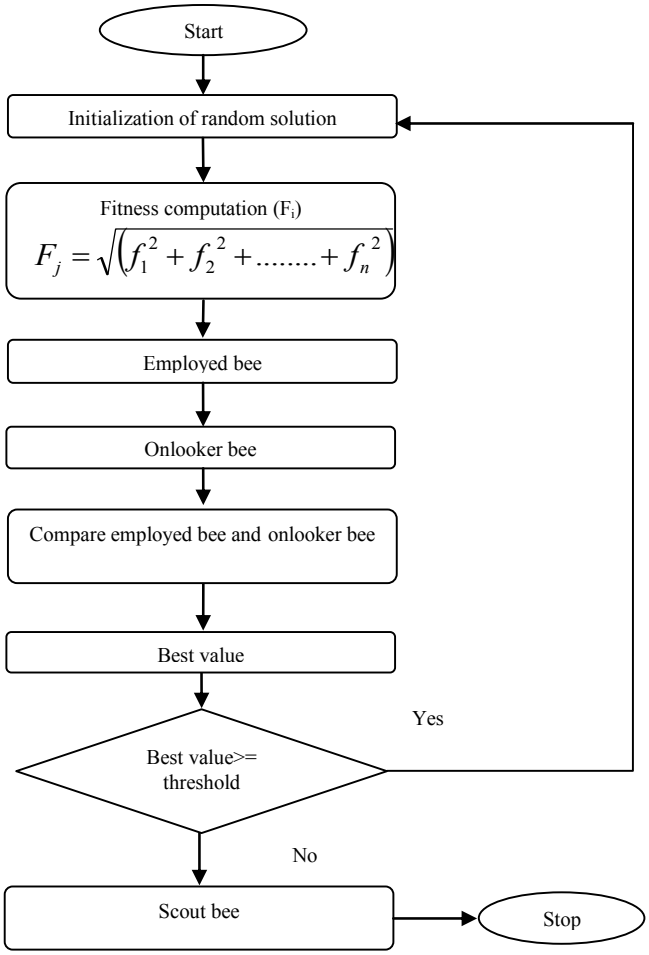

Figure 1 Proposed Algorithm

In the artificial bee colony algorithm, the artificial bee is arbitrarily classified into three diverse categories such as employed bee, onlooker bee and scout bee with each category of bees equipped with distinct and unique characteristics. A bee who is an expert in utilizing its food source to the full gets the name 'employed bee' and he is entrusted with the task of communicating this piece of data with the onlooker bee. The onlooker bee waits in the hive enjoys the dance of employed bee and selects a food source according to the prospect resonance of the food source. Thus superb food sources are engaged to onlooker bee. In the accompany phase, scout bee hunts for the fresh food source and on successfully achieving it by locating a fresh food source it metamorphoses into an employed bee. This newly converted employed bee performs the tasks of utilizing the whole food source, in the process gets rid of its position and eventually reincarnates as the scout bee. Thus, we are able to comprehend that the function of scout bee is investigation and the task of employed and onlooker bee is utilization.

4.1.1. Initialization of random solution:

At the outset, the population of the food source is initialized. Subsequently, the population goes through cycles with 4 fundamental phases such as revising of viable solutions by employed bee, choice of realistic solution by onlooker bee, reviewing viable solution by onlooker bee and evading optimal solution by scout bee. In our proficient method the food sources are tasks which are performed under every task in an arbitrary way.

4.1.2. Fitness computation:

With the created food source we have to determine the amount of the nectar available in the food source. Therefore, we proceed to estimate the fitness value with the available amount of food source. This fitness function decides the pertinent food source which proceeds to the following generation. If the fitness values of optimal solution are lower, then the probability to stay alive in the ensuing generation is greater. The fitness is estimated by the equation:

$$F_j = \sqrt{\left(f_1^{\,2} + f_2^{\,2} + \ldots\ldots + f_n^{\,2}\right)} \qquad (1)$$

where $f_1, f_2, \ldots f_n$ are the food sources.

4.1.3. Employed bee phase:

Employed bee is entrusted with the task of hunting for the food source and conveying the valuable data to the onlooker bee. In this phase, every employed bee possesses its own unique solution and adapts it to turn out the fresh solution.

4.1.4. Onlooker bee phase:

When the employed bee completes its duty of local investigation, it conveys the nectar information of the food source to the onlooker bee, which proceeds to choose a fresh food source in the accepted way. When the possibility of the chosen constraint is determined, number of onlooker bees is calculated. Thereafter, fresh solutions ($V_{i,j}$) are engendered for the onlooker bees from the solutions ($x_{i,j}$) in accordance with the probability value (P_j). Subsequently, the fitness function is determined for the fresh solution. Then the greedy choice task is performed with a view to choose the best parameter.

The adaptation is performed as per the following equation:

$$V_{i,j} = x_{i,j} + \phi_{ij}\left(x_{i,j} - x_{k,j}\right) \qquad (2)$$

Where, k and j are m arbitrary chosen indexes, ϕ arbitrarily generated number in the range [-1, 1] and $V_{i,j}$, the fresh value of the j location. Thereafter the fitness value is estimated for each freshly created population constraints of food sources. From the determined fitness value of the population, best population parameter is chosen.

4.1.5. Comparison of employed and onlooker bee:

At this point, the values of the employed and onlooker bee are contrasted with each other and the best value is saved in the "best value". Thereafter the "best value" is compared with the threshold value and if the "best value" does not exceed the threshold value then the scout bee is generated. If it exceeds the threshold value, then the task is replicated from the generation of fresh solutions.

4.1.6. Scout bee phase:

In the ABC algorithm, if the quality of the solution does not get better than the threshold value limit, then the food source is treated as excess and the relative employed bee turns into a scout bee. Then the discarded restraints for the scout bees are estimated. If any such discarded restraints still persist, then they are substituted with fresh constraints located by scouts and the fitness value is estimated by means of the fitness equation. Thereafter memorize the best constraints generated till now are committed to memory. Subsequently, the iteration is incremented and the task is repeated till the stopping standard is attained. At last, the decreased make span of the processing period is found out.

4.2. Genetic Algorithm

The workability of genetic algorithms (GAs) is based on Darwinian's theory of survival of the fittest. Genetic algorithms (GAs) may contain a chromosome, a gene, set of population, fitness, fitness function, breeding, mutation and selection. The genetic algorithms performance is largely influenced by crossover and mutation operators. The block diagram representation of genetic algorithms (GAs) is shown in Fig.1.The GA is a stochastic investigation technique which boosts the natural choice procedure. The GA takes into account a population of solutions, which are known as chromosomes. Each and every solution is symbolized with an encoding method which enables deciphering a solution into a series of genes forming part of a chromosome.

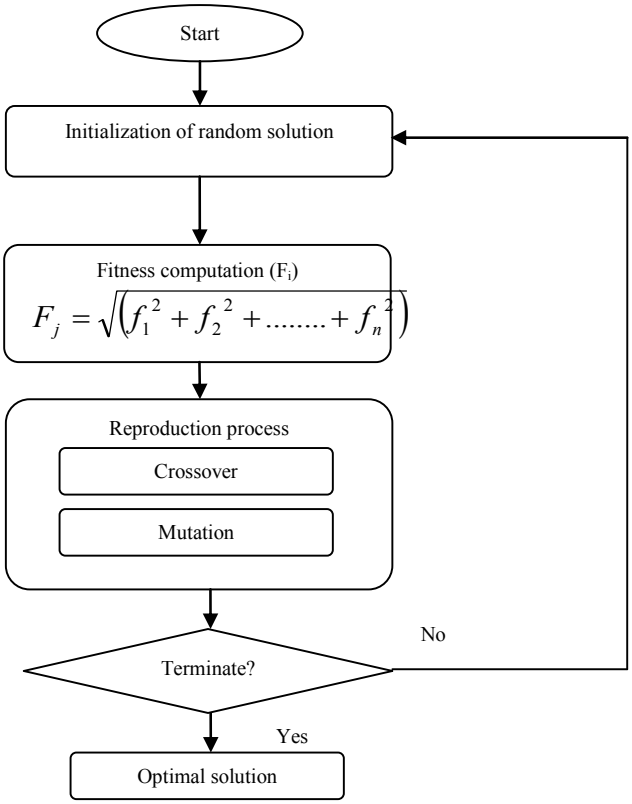

Fig 2: flow chart of Genetic algorithm

Each iteration of GA comprises many operators. From among them, reproduction, crossover and mutation are the most widespread ones. The reproduction operator chooses the finest fit chromosomes to the following generation. Crossover enables solutions to communicate data from two discretely selected parents so as to create one or more offspring which houses certain amalgamation of genes from the parents. The mutation operator discretely adapts certain genes available in a chromosome. In our task, we make use of the immigration operator in place of the mutation operator. The immigration operator generates a minimal number of fresh solutions with the aid of the process employed to build the preliminary population.Genetic algorithms (GAs) begin with a set of solutions represented by chromosomes, called population. Solutions from one population are taken and used to form a new population, which is motivated by the possibility that the new population will be better than the old one. Further, solutions are selected according to their fitness to form new solutions, that is, offsprings. The above process is repeated until some condition is satisfied.

5. Result and discussion

Here, we proceed to assess the outcome of our projected technique, by investigating the outcome in MATLAB software. This part focuses on the solitary machine which performs 5 tasks and every task having 8 processes each. An identical task can be accomplished on multiple machines. In our novel method we have carried out the testing with 5 machines. The overall time of every task for solitary machine is furnished below.

Table 1: Time allocation for job process in single machine

process job	1	2	3	4	5	6	7	8
1	9.1012	4.1187	0.7445	5.8405	0.2548	8.7072	1.7847	6.9213
2	4.9044	2.7368	8.5438	0.8122	1.7514	8.7637	2.7263	2.8699
3	0.1177	5.6355	5.1467	2.7246	2.4305	2.4627	5.6133	5.9735
4	8.7275	8.5019	8.6386	7.5224	6.614	7.603	7.2683	1.9597
5	5.7145	9.8054	0.3585	9.5567	4.5662	4.9609	4.9428	4.436

Table 2: ABC and GA output with different parameters in single machine

Parameters	GA output	ABC output
Total processing time for 5 jobs	201.8623	201.8623
Processing time for 5 jobs by algorithm	150.5225	55.5293
Total time saved	51.3398	146.333
Number of process saved	13	27
Efficiency	32.5	67.5

The table shown above illustrates the output part of ABC algorithm and GA algorithm for 6 diverse constraints. At first, the total processing time for the entire five jobs are determined physically and it is allocated as identical for both the algorithms. Table 2 makes it obvious that there is reduced processing interval in the case of computing by ABC algorithm in relation to GA, for the entire 5 jobs. Therefore the total time saved tends to be higher during the estimation by ABC procedure and hence the number of tasks saved also increases. The captioned table depicts the output for solitary machine efficiency.

Table 3: ABC and GA output with different parameters in 5 machines

Parameters	GA output	ABC output
Total processing time for 5 machines	1009.3115	1009.3115
Processing time for 5 machines by algorithm	725.6125	277.6465
Total time saved	256.699	731.665
Number of process saved	65	135

If the total number of machine is 5 then the completion of job by ABC algorithm can be done completed by 2 machines and the last single process is done in the 3rd machine.

If the total number of machine is 5 then the completion of job by GA algorithm can be done completed by 3 machines and the last seven processes are done in the 4th machine. This represents GUI for MATLAB output and it furnishes the input and output of our innovative technique. At first, the input furnished as the overall processing interval for the entire 5 processes is 201.8623sec determined by summing up the individual run times. The output part delineates 4 various constraints for both ABC and GA technique. In the case of ABC output, the total processing time (TPT) is taken as 55.5293sec, time saved (TS) is 146.33sec, total process saved (TPS) is 27 and efficiency (E) is 67.5. Taking all these constraints, the output of GA is estimated whose value is observed to be as follows: TPT, 150.5225sec, TS, 51.3398, TPS, 13 and E, 32.5. An analysis of GUI output makes it crystal clear that the ABC achieves superior efficiency turning out a superior performance in relation to that of GA.

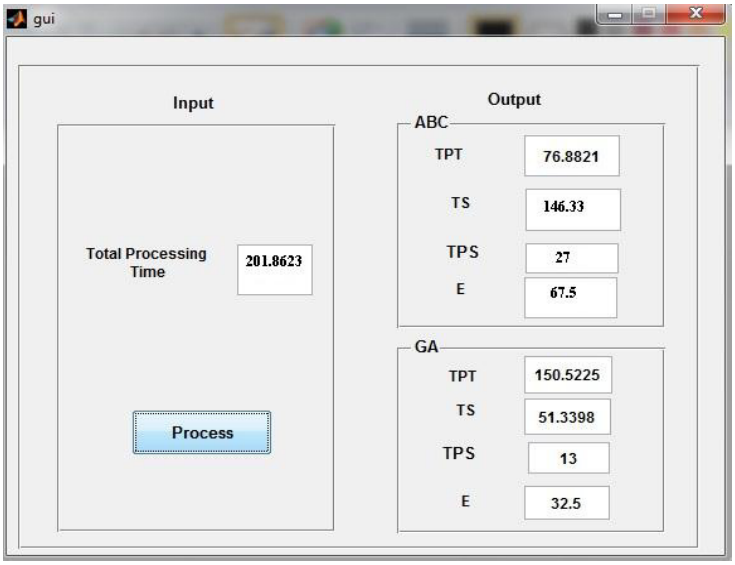

Fig 3: MATLAB output for ABC and GA

This graph associates fitness with various iterations for ABC and GA. It is evident from graph that the fitness value for GA does not change disregarding any perceptible alteration in the iteration. However, in the case of ABC, as the iteration goes up, it leads to the decrease in the fitness value and at last the solution related to the lower fitness is achieved in the distinct iteration. Therefore, the lesser fitness value is taken into account and the analogous solution is considered for the task.

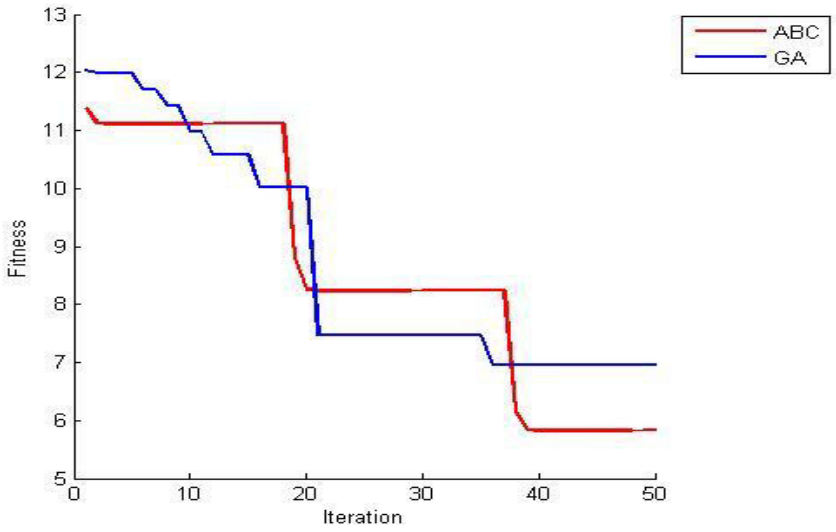

Fig 4: Output performance of ABC and GA

6. Conclusion

In our research work, we have effectively employed ABC algorithm to reduce the make span of the job scheduling task to the minimum. On analysis and contrast of the outcomes with those of GA, it is unequivocally established that ABC algorithm is competent to achieve amazing outcomes. We have also analyzed and contrasted the run time of the two processes according to the overall processing interval of the CPU for finishing the aggregate number of iteration. In this connection, we have made use of 5 machines with every machine executing 5 jobs and every job, in turn, performing 8 processes. Thus, in all, there are 40 processes in a solitary machine. Hence, we are cheered to note that the total processing interval is decreased in ABC, prompting us to assert that the number of machines employed for every process can be decreased. The upcoming research in this regard will be founded on defining the job wherein we intend to investigate the employment of a bigger set of constraints for our process which will be analyzed and contrasted with parallel top algorithms and our aim also will focus on putting JSSP to investigation in cases where ‘‘job interrupt’’ is allowed.

Reference

[1] Xiaoping Li, Yi Zhang, Adaptive hybrid algorithms for the sequence-dependent setup time permutation flow shop scheduling problem, IEEE T Auto Sci Eng, 9 (2012), 578-595.

[2] Li-Ning Xing, Ying-Wu Chen, Peng Wang, Qing-Song Zhao, Jian Xiong, A knowledge-based ant colony optimization for flexible job shop scheduling problems, Appl S Comput, 10 (2010), 888-896.

[3] Ye Li, Yan Chen, A genetic algorithm for job-shop scheduling, J Softw , 5 (2010), 269-274.

[4] Wenbin Gu, Dunbing Tang, Kun Zheng, Minimizing make span in job-shop scheduling problem using an improved adaptive particle swarm optimization algorithm, Proc Cont Decis Conf, Taiyuan, (2012), 3189-3193.

[5] Amel Yahyaoui, Nader Fnaiech, Farhat Fnaiech, A suitable initialization procedure for speeding a neural network job-shop scheduling, IEEE T Ind Elect, 58 (2011), 1052-1060.

[6] Kagan Gokbayrak, Omer Selvi, Service time optimization of mixed-line flow shop systems, IEEE T Auto Cont, 55 (2010), 395-404.

[7] Wei Sun, Ying Pan, Xiaohong Lu, Qinyi Ma, Research on flexible job-shop scheduling problem based on a modified genetic algorithm, J Mech Sci Technol, 24 (2010), 2119-2115.

[8] De-ming Lei, Minimizing make span for scheduling stochastic job shop with random breakdown, Journal of applied mathematics and computation, Vol. 218, pp. 11851-11858, 2012.

[9] Jie Gao, Linyan Sun, Mitsuo Gen, A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems, J Comput Oper Res, 35 (2008), 2892-2907.

[10] Ren Qing-dao-er-ji, YupingWang, A new hybrid genetic algorithm for job shop scheduling problem J Comput Oper Res, 39 (2012), 2291-2299.

[11] Javid Taheri, YoungChoonLee, AlbertY.Zomaya, HowardJaySiegel, A bee colony based optimization approach for simultaneous job scheduling and data replication in grid environments, J Comput Oper Res, 40 (2013), 1564-1578.

[12] Rui Zhang, An artificial bee colony algorithm based on problem data properties for scheduling job shops, J Proc Eng, 23 (2011), 131-136.

[13] Ravi Kumar Jatoth, A.Rajasekhar, Speed control of PMSM by hybrid genetic artificial bee colony algorithm, Proc Commu Con Comput Technol, India, 241-246.

[14] D.Y. Sha, Hsing-Hung Lin, A multi-objective PSO for job-shop scheduling problems, Expert Syst Appl, 37(2010), 1065-1070.

[15] D. Hajinejad, N. Salmasi, R. Mokhtari, A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem, J Sci Iran, 18 (2011) 759-764.

[16] Anan Banharnsakun, Booncharoen Sirinaovakul, TiraneeAchalakul, Job shop scheduling with the best-so-far ABC, J Eng Appl Artifi Intell, 25 (2012), 583-593.

[17] Jun-Qing Li, Quan-Ke Pan, M. Fatih Tasgetiren, A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities, J Appl Math Model, 38 (2013), 1111–1132.

[18] Antonin Ponsicha, Carlos A. Coello Coello, A hybrid differential evolution - tabu search algorithm for the solution of job-shop scheduling problems, Journal of applied soft computing, Vol. 13, pp. 462-474, 2013.

[19] T.C. Wong, S.C. Ngan, A comparison of hybrid genetic algorithm and hybrid particle swarm optimization to minimize makespan for assembly job shop, Appl S Comput, 13 (2013), 1391-1399.

[20] Rui Zhang, ShijiSong, ChengWu, A hybrid artificial bee colony algorithm for the job shop scheduling problem,, Int. J. Prod Econ, 18(2011), 759-764.